Project 2

By

Valerie A. Rivera

&

Armando Herrera

University of Texas at Rio Grande Valley

CSCI 6366: Data Mining

Fall 2019

October 20, 2019

Introduction:

Our assignment involved the MNIST datasets,which consists of training set and testing sets. Our task consisted of utilizing the training set to train the SVM classifier to its best status. Then, use the testing dataset to conduct classification. In the following paper we will be doing the following:

Task 1: Report classification accuracy on the testing dataset.

Task 2: Find out the testing samples which are not correctly classified. Generate more samples to challenge the SVM classifiers. Report the accuracy.

Therefore this report will consist of describing the design, settings, and the results of the assignment. The first part of the paper will describe Valerie Rivera section of the project. The second part will describe Armando Herreras section of the project.

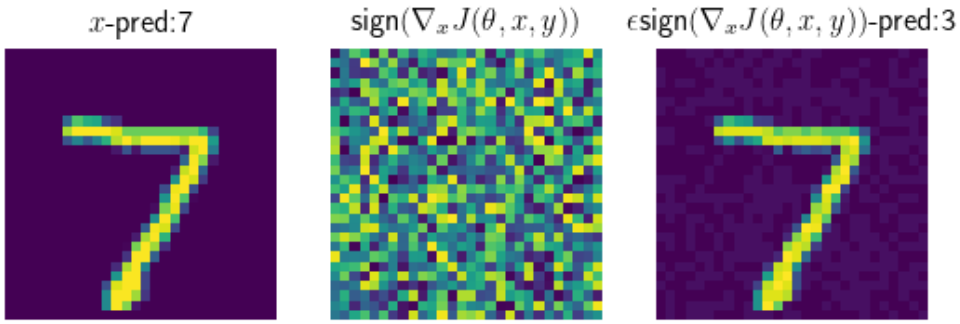
Github Link: https://github.com/anhydrous99/SVM_GAN

Armando Herreras Section of the Project

Design:
An adversarial attack used to generate samples is called *Fast Gradient Sign Attack (FGSM)* and is described in a paper titled, “Explaining and Harnessing Adversarial Examples”. The paper shows that by calculating the gradient, $\nabla_x J(\theta, x, y)$, through a model, neural network or SVM, of an image, through backpropagation, and by adding the sign of the gradient multiplied by a hyper-parameter, ϵ, we can decrease the accuracy of models that relies heavily on linear elements.
$f(x) = x + \epsilon sign(\nabla_x J(\theta, x, y))$

Settings:
The goal for the epsilon hyper-parameter was to set low enough for it to produce samples with barely recognizable changes and still fool the SVM. An epsilon value of 0.08 was chosen.

Results:
The result was a PyTorch system that brings down the classification accuracy to 10.92%.



The leftmost image is of a correctly classified image of a seven classified as a seven. The middle image is of the gradient map created from back-propagating through the SVMs. The rightmost image is of the perturbed image which was incorrectly classified as a three instead of what it is, a seven.

Valerie Rivera’s Section of the Project

Design:

- Mathematical:

The mathematical design is a Support Vector Machines (SVM). We begin with a linear Support Vector Machine. It is important to note that because an SVM is a binary classifier, it classifies for, against, or between classes. In our specific design, we created an SVM for every class. My partner and I decided instead of doing many dot operations to do a single matrix operation for every SVM. Therefore, we combined every SVM into a single weight and a biases matrix.

- Linear SVM classifier $\vec{w} \cdot \vec{x} + b$
- Our multi-class SVM classifier $\bar{w} \cdot \vec{x} + \vec{b}$

Implementation:

Two implementations introduced into our project is the loss function, which is cross-entropy. The optimization algorithm used to minimize our weight and biases is known as stochastic gradient descent.

We created a weight PyTorch matrix and a biases PyTorch vector, which holds a distinctive characteristic. This characteristic comes into play when one performs operations on a PyTorch object. The PyTorch object remembers the processes, which are later used to backpropagate through our matrix multiplication, our addition operation, and our loss function. This helps in turn to minimizes the error of the SVM. We are, therefore training it to classify MNIST.

Settings:

The optimal settings that we found were with the learning rate at 0.007 and training the SVM for six epochs. This means that we trained the network over all of the data six times. When the epoch would finish, the input data were randomly shuffled. All the input images were scaled from between 0 to 255 to -1 to 1, which is a range of 8-bit integers, and converted to single-precision floating-point numbers.

Results:

We achieved an accuracy of 89.42 percent in our MNIST testing dataset. We saved the SVM to a file for future use.

Conclusion

We learned to build an SVM that classifies images of handwritten numbers.
The limitations of this project were our time limitation and the limitation of full knowledge of SVMS. Our hope in the future is to implement the kernel trick. This implementation would help to classify not only linear but also non-linear data.